02

Errores

Transcripción

[00:00] Sean bienvenidos a la segunda clase de su curso de Java, entiendo excepciones. Bueno, nuevamente, lo prometido es deuda y vamos a ver ahora sí lo que son errores y excepciones. Básicamente son lo mismo. A modo de repaso, vamos aquí a ver la última parte donde nos quedamos en la clase anterior.

[00:25] Ya habíamos hablado de la pila de ejecución ya conocemos la pila de ejecución. Hemos visto en el código cómo es que se van ejecutando poco a poco los métodos, como el método main, sirve de soporte para método 1 y a la vez método 1, sirve de soporte para método 2. Hasta que métodos no finalice, método 1 no puede salir de la pila de ejecución.

[00:46] Y de igual forma, hasta que método 2 y método 1 no finalicen, main no puede salir de la pila de ejecución. Entonces, ¿qué es lo que llamamos un error en el código? Un error es cuando de pronto llega un suceso inesperado en método 2 o en método 1 que impide que terminen su ejecución.

[01:10] Vamos a verlo en el código para hacerlo un poco más práctico. Está aquí. Y bueno, este es el código que ya teníamos antes, nuestra clase flujo y lo que vamos a hacer ahora es usar nuestro proyecto anterior de polimorfismo y no precisan descargarlo. Ojo, no se preocupen, lo que quiero hacer aquí es provocar un error a propósito y para esto voy a usar mi clase cuenta.

[01:36] ¿Recuerdan aquí la clase cuenta? Cuenta corriente o cuenta de ahorro. Recuerden que hicimos, pues herencia usando el objeto cuenta. Y yo aquí

quiero causar un error. Yo quiero causar un error a propósito. El error que voy a causar va a ser un error bien, bien simple, pero que a veces sucede. Yo voy a inicializar cuenta corriente como null.

[02:04] ¿Qué quiere decir esto? Que cuenta corriente CC, no va a tener ningún lugar en la memoria en mi programa. Por lo tanto, cada referencia que apunte aquí va a apuntar a nul que no tiene nada. Por lo tanto, ¿ustedes creen que el método deposita y transfiere funcione? Ahora, dense cuenta que el código compila cuenta corriente igual null, va a dar un error, pero compila bien.

[02:32] Vamos a ver, primero vamos paso a paso, vamos a ver qué errores de qué va a dar. Vamos a ejecutar aquí, cuenta, y vemos aquí en rojo. ¿Qué dice? "Exception in thread". Thread es hilo, es el hilo ejecución que ha estado siguiendo, el flujo de ejecución, podemos entenderlo de esa forma, main. Entonces, en el flujo ejecución de ejecución main, hubo un null Java.lang.NullPointerException.

[03:01] No se preocupen mucho ahora por el nombre y aquí nos dice "at bytebank. TestCuenta.main(TestCuenta.java:8)" Aqui. ¿Qué quiere decir esto? Cuando él llamó al método deposita él dio un NullPointerException. Y pero es raro, porque el código está compilando. Sí, el código compila, incluso si vamos aquí a la vista de Navigator, vemos pues que el archivo punto class aquí, perdón, no es aquí.

[03:33] Es aquí en cuenta. Vamos aquí a bin, bytebank. Vemos que nuestro TestCuenta.class existe, incluso si yo lo eliminó aquí. ¿Puedo eliminar este punto class? Okay, lo borro y acá hago una actualización cualquiera, lo guardo automáticamente, él genera uno nuevo, entonces el código sí está compilando, este código sí se está traduciendo a byte code.

[04:01] Y ese byte code sí está siendo capaz de ser leído por la Java Virtual Machine. ¿Entonces, dónde está el problema? Bueno, el problema está en la referencia nula. Nuevamente, como les dije, yo estoy haciendo referencia a un

lugar nulo en la memoria entonces no existe. Por lo tanto, el método deposita no tiene a quién hace referencia, no tiene quién depositarle.

[04:21] Entonces, ahí es donde él suelta esa excepción, con ese nombre un poco raro, NullPointerException. En el caso de de Java los errores tienen nombre. En este caso es NullPointerException, que ya nos da como que una idea de por dónde puede puede estar el error.

[04:42] Entonces aquí, volviendo pues a nuestra presentación, imaginémonos que la excepción puede ser como una especie de bomba que llega a tu código y tu código no sabe cómo desactivar esa bomba, por ejemplo. Déjenme ver aquí otro ejemplo. Puede ser también un error de este tipo, por ejemplo, puedo declarar un entero, número, que va a ser igual a cero.

[05:18] Perfecto, declaro mi entero igual a cero y un entero resultado que va a ser igual, digamos a 30, entre num, que es iguala cero. Perfecto. Aquí vemos que él quiere dividir por cero en Java y la mayoría lenguaje de programación es algo que no se puede. No pueden dividir por cero, entonces aquí le vamos a dar un System.out.println.

[05:520] System.out.println, está más abajo, ln, perfecto. Y le vamos a decir que nos imprima el resultado. Perfecto y ahora si ejecutamos, guardamos otra vez. Ejecutamos aquí. Dio otro error, mira: "Java.lang.ArithmeticException / by zero". ¿Qué significa? Has querido dividir entre cero, y acá nos da el detalle de dónde sucedió.

[06:32] Vamos a entrar a ver un poco más de detalle esto, pero básicamente lo que quiero que vean es cada tipo de error que tenemos en Java tiene un nombre diferente, una clasificación diferente de error, primera cosa que tenemos que notar en las excepciones. Voy a copiar este mismo error para regresar a nuestro flujo.

[06:51] Vengo aquí al flujo. Y por ejemplo, pues en el método 2, aquí adentro, yo voy a hacer esa misma esa misma jugada. Incluso no 30, sino el índice. Puede

ser cualquier número, ningún lenguaje de programación consigue dividir entre 0. Sí, acá nuevamente le doy play, sí, quiero guardarlo, nuevamente vemos claramente que comienza inicio main, inicio método 1, inicio método 2. Imprime 1 que está aquí, imprime 1 y excepction.

[07:31] ¿Y qué pasó con Fin método 2? ¿Qué pasó con fin método 1? ¿Qué pasó con Fin de main? Vamos a averiguarlo en el próximo video.